Artificial Intelligence Project--RLE and MIT Computation Center
Memo 37 --

15p

NOT AVAILABLE

Some Identities Concerning the Function subst [x; y; z]

by Lewis M. Norton

submitted January 19, 1962
as a term paper for 6.539
under Professor John McCarthy

Revised March 6, 1962

Some Identities Concerning the Function
subst [x; y; z]

The purpose of this paper is two-fold; 1) to explore the use of recursion induction in proving theorems about functions of symbolic expressions, in particular
2) to investigate thoroughly the algebraic properties of the LISP function subst [x; y; z] by this method. The main result is embodied in Theorem 8.

For reference, the function is defined by:

subst [x; y; z] = [atom[z] → [eq[z;y] → x; T → z]; T → cons [subst[x;y;car[z]];subst[x;y;cdr[z]]]];

where x and z are S-expressions, and y is an atomic symbol. We will make the agreement that y ∼null[y].

In the work of this paper, dots (.....) will often appear in the functional expressions. Such dots will represent superfluous material only; e.g. if we know atom [z] is true we may write:
subst[x;y;z] = [atom[z] → [eq[z;y] → x; T → z]; T → .....].

Reference will be made to theorems proved on page 25 of Memo 31-- A Basis for a Mathematical Theory of Computation by John McCarthy, as well as to expressions for car[x*y], cdr[x*y], and similar expressions on that page.

We start by proving five lemmas:

Lemma 1: If ∼ atom [u], then car[subst[x;a;u]] = subst[x;a;car[u]] and cdr[subst[x;a;u]] = subst[x;a;cdr[u]].

Proof: car[subst[x;a;u]]
= car[atom[u] → ...; T → cons[subst[x; a; car[u]]; subst[x;a;cdr[u]]]]
which, employing the hypothesis, reduces to
= car[cons[subst[x;a;car[u]];subst[x;a;cdr[u]]]]
= subst[x;a;car[u]].
A similar computation establishes the equality for cdr[subst[x;a;u]].

Q. E. D.

Lemma 2: If ∼atom[u] (which insures the existence of both forms), then subst[x;a;cons[car[u];cdr[u]]] =
cons[subst[x;a;car[u]];subst[x;a;cdr[u]]].

Proof: The left side equals subst[x;a;u]
while the right side equals (by Lemma 1)
cons[car[subst[x;a;u]]; cdr[subst[x; a; u]]]
= subst[x;a;u]

Q. E. D.

Lemma 3: If $\sim$ atom [u] (which insures the existence of both forms), then subst[x;a;cons[subst[y;b;car[u]];subst[y;b;cdr[u]]]]
= cons[subst[x;a;subst[y;b;car[u]]];subst[x;a;subst[y;b;cdr[u]]]].

Proof: Since by hypothesis $\sim$ atom[u], then [$\sim$ atom[subst[y;b;u]] follows immediately from the definition of the function subst[x;y;z], since $\sim$ atom[cons[a;b]] = T is a basic LISP identity. Thus, applying Lemma1, we may let subst[y;b;u] take the place of u in the proof of Lemma 2, and Lemma 3 follows directly, using Lemma 1 once again.

Q. E. D.

Lemma 4: subst[x;a:NIL] = NIL

Proof: subst[x;a;NIL] = [atom[NIL] → [eq[NIL;a] → x; T → NIL];T →...] and since $\sim$ eq[a;NIL] by convention, this reduces to NIL.

Q. E. D.

Lemma 5: If $\sim$ atom[u], then car[u] has meaning, and subst[x;a; cons[car[u];NIL]] = cons[subst[x;a;car[u]];NIL].

Proof: The left-hand side equals:

[atom[cons[car[u];NIL]] → ...; T → cons[subst[x;a;
car[cons[car[u];NIL]]];subst[x;a;cdr[cons[car[u];NIL]]]]]
= cons[subst[x;a;car[u]];subst[x;a;NIL]]
= cons[subst[x;a;car[u]];NIL] by Lemma 4.

Q. E. D.

We now prove a trivial theorem:

Theorem 1: subst[a;a;u] = u

Proof: subst[a;a;u] =
[atom[u] → [eq[u;a] → a; T → u]; T → cons[subst[a;a;car[u]];
subst[a;a;cdr[u]]]]

Employing the principle of recursion induction, we consider
f[a;u] = [atom[u] → [eq[u;a] → a; T → u]; T → cons[f[a;car[u]];
f[a;cdr[u]]]]

We now note that u may be written
u = [atom[u] → [eq[u;a] → a; T → u]; T → cons[car[u];cdr[u]]]

Therefore both sides of the equation satisfy the equation f[a;u].

Q. E. D.

Some comment should be made noting that f[a;u] indeed converges, but other than so noting, such comments will be postponed.

We now formally define the condition that there be no occurrences of the atomic symbol a, ($\sim$ null[a]), in the S-expression y by the formula

$$free[a;y] = [atom[y] \rightarrow \sim eq[y;a];\ T \rightarrow free[a;car[y]] \wedge free[a;cdr[y]]].$$

The main reason for introducing this concept is embodied in the following lemma:

<u>Lemma 6:</u> If $free[a;y]$, then $subst[x;a;y] = y$.

Proof: $subst[x;a;y] = [atom[y] \rightarrow [eq[y;a] \rightarrow x;\ T \rightarrow y];\ T \rightarrow cons[subst[x;a;car[y]];subst[x;a;cdr[y]]]]$

$= [atom[y] \rightarrow y;\ T \rightarrow cons[subst[x;a;car[y]];subst[x;a;cdr[y]]]]$

employing the hypothesis.

Now $y = [atom[y] \rightarrow y;\ T \rightarrow cons[car[y];cdr[y]]]$.

Both equations satisfy the functional equation

$$f[x;a;y] = [atom[y] \rightarrow y;\ T \rightarrow cons[f[x;a;car[y]];f[x;a;cdr[y]]]].$$

Q. E. D.

The next theorem states that, with certain restrictions, the order of substitution is irrelevant.

<u>Theorem 2:</u> If $\sim eq[a;b]$, $free[a;y]$, and $free[b;x]$, then

$$subst[x;a;subst[y;b;u]] = subst[y;b;subst[x;a;u]].$$

Proof: $subst[x;a;subst[y;b;u]]$

$= subst[x;a;[atom[u] \rightarrow [eq[u;b] \rightarrow y;\ T \rightarrow u];\ T \rightarrow cons[subst[y;b;car[u]];subst[y;b;cdr[u]]]]]$

$= [atom[u] \rightarrow [eq[u;b] \rightarrow subst[x;a;y];\ T \rightarrow subst[x;a;u]];\ T \rightarrow subst[x;a;cons[subst[y;b;car[u]];subst[y;b;cdr[u]]]]]$

$= [atom[u] \rightarrow [eq[u;b] \rightarrow y;\ T \rightarrow subst[x;a;u]];\ T \rightarrow cons[subst[x;a;subst[y;b;car[u]]];subst[x;a;subst[y;b;cdr[u]]]]]$ *

$= [atom[u] \rightarrow [eq[u;b] \rightarrow y;\ T \rightarrow [atom[u] \rightarrow [eq[u;a] \rightarrow x;\ T \rightarrow u];T \rightarrow \ldots]];T\ cons[subst[x;a;subst[y;b;car[u]]];subst[x;a;subst[y;b;cdr[u]]]]]$

$= [atom[u] \rightarrow [eq[u;b] \rightarrow y;\ T \rightarrow [eq[u;a] \rightarrow x;\ T \rightarrow u]];\ T \rightarrow cons[subst[x;a;subst[y;b;car[u]]];subst[x;a;subst[y;b;cdr[u]]]]]$

and since the hypothesis $\sim eq[a;b]$ implies the conditions $eq[u;b]$ and $eq[u;a]$ are mutually exclusive, this can be written as:

$= [atom[u] \rightarrow [eq[u;a] \rightarrow x;\ T \rightarrow [eq[u;b] \rightarrow y;\ T \rightarrow u]];\ T \rightarrow cons\ [subst[x;a;subst[y;b;car[u]]];subst[x;a;subst[y;b;cdr[u]]]]]$

but $[eq[u;b] \rightarrow y;\ T \rightarrow u]$ is just the expression for $subst[y;b;u]$ given that u is atomic, so we have finally:

$= [atom[u] \rightarrow [eq[u;a] \rightarrow [eq[u;a] \rightarrow x;\ T \rightarrow subst[y;b;u]];\ T \rightarrow cons[subst[x;a;subst[y;b;car[u]]];subst[x;a;subst[y;b;cdr[u]]]]]$

---

* "employing Lemmas 3 and 6

Applying the same procedure to the right side of the equation, we find:

subst[y;b;subst[x;a;u]] =
= [atom[u] → [eq[u;a] → x; T → subst[y;b;u]]; T →
cons[subst[y;b;subst[x;a;car[u]]];subst[y;b;subst[x;a;cdr[u]]]]]

as in the first three steps of this proof. Again we have made use of Lemmas 3 and 6.

Clearly then, both sides of the equation satisfy the functional equation:

f[a;b;x;y;u] = [atom[u] → [eq[u;a] → x; T → subst[y;b;u]]; T →
cons[f[a;b;x;y;car[u]];f[a;b;x;y;cdr[u]]]]

and the principle of recursion induction yields the identity.

Q. E. D. Theorem 2

Our next result states that in a certain sense, the operation of substitution is transitive.

Theorem 3: If free[a;u], then

subst[x;a;subst[a;y;u]] = subst[x;y;u]

Proof: subst[x;a;subst[a;y;u]] =
subst[x;a;[atom[u] → [eq[u;y] → a; T → u]; T → cons[subst[a;y;car[u]];
subst[a;y;cdr[u]]]]]
= [atom[u] → [eq[u;y] → subst[x;a;a];T → subst[x;a;u]]; T →
subst[x;a;cons[subst[a;y;car[u]];subst[a;y;cdr[u]]]]]

Now subst[x;a;a] = [atom[a] → [eq[a;a] → x; T → ...]...]
= x so we have, using this and Lemma 6,

which is applicable by our hypotheses:

= [atom[u] → [eq[u;y] → x; T → u]; T →
cons[subst[x;a;subst[a;y;car[u]]];subst[x;a;subst[a;y;cdr[u]]]]]
(using Lemma 3 at the end)

This form suggests considering the functional equation

f[x;a;y;u] = [atom[u] → [eq[u;y] → x; T → u]; T →
cons[f[x;a;y;car[u]];f[x;a;y;cdr[u]]]]

Now subst[x;y;u] = [atom[u] → [eq[u;y] → x; T → u]; T →
cons[subst[x;y;car[u]];subst[x;y;cdr[u]]]]

so it also satisfies the functional equation, thus proving the desired identity.

Q. E. D.

Corollary: If free[a;u], then

subst[y;a;subst[a;y;u]] = u , an intuitively obvious identity.

Proof: By Theorem 3, with eq[x;y]

subst[y;a;subst[a;y;u]] = subst[y;y;u]
= u (by Theorem 1) Q. E. D.

In the case of the corollary, it is not difficult to compute an identity without the hypothesis. In other words, the corollary could be proved as a special case of

<u>Theorem 4:</u> subst[y;a;subst[a;y;u]] = subst[y;a;u]

for the corollary follows directly from Theorem 4 and Lemma 6.

To prove Theorem 4:

<u>To prove Theorem 4:</u>

subst[y;a;subst[a;y;u]] =
= subst[y;a;[atom[u] → [eq[u;y] → a;T → u]; T → cons[subst[a;y;car[u]];
subst[a;y;cdr[u]]]]]
= [atom[u] → [eq[u;y] → subst[y;a;a]; T → subst[y;a;u]]; T →
subst[y;a;cons[subst[a;y;car[u]];subst[a;y;cdr[u]]]]]

and since subst[y;a;a] = y, as noted in the proof of Theorem 3, we have

= [atom[u] → [eq[u; y] → y; T → [atom[u] → [eq[u;a] → y; T→u];T →..]]; T →
subst[y;a;cons[subst[a;y;car[u]];subst[a;y;cdr[u]]]]]
= [atom[u] → [eq[u;y] → u; T → [eq[u;a] → y; T → u]]; T →
cons[subst[y;a;subst[a;y;car[u]]];subst[y;a;subst[a;y;cdr[u]]]]]
(by Lemma 3)
= [atom[u] → [eq[u;a] → y; T → u]; T →
cons[subst[y;a;subst[a;y;car[u]]];subst[y;a;subst[a;y;cdr[u]]]]]

Now subst[y;a;u] = [atom[u] → [eq[u;a] → y; T→u];T→cons[subst[y;a;car[u]];
subst[y;a;cdr[u]]]]

so that both forms satisfy

f[a;y;u] = [atom[u] → [eq[u;a] → y; T → u]; T → cons[f[a;y;car[u]];
f[a;y;cdr[u]]]]

Q. E. D.

At this point we introduce the concatenation x*y of two lists x and y, as defined on page 25 of the report.mentioned earlier. For convenience, the definition and two identities from that paper are reproduced below:

x*y =[null[x]→y; T → cons[car[x];cdr[x]*y]]
car[x*y] = [null[x] →car[y]; T → car[x]]
cdr[x*y] = [null[x] →cdr[y];T → cdr[x]*y]

In the work done so far, we have always made the tacit assumption that our expressions are well-defined. For example; whenever subst[x;y;z] has appeared in the statement of a theorem, y has automatically been assumed to be atomic. (Also we have made the blanket

assumption that ~null[y].) Since the concatenation is only defined for true lists, we will make similar tacit assumptions. For instance, in the statement of the next theorem u and v are automatically assumed to be true lists.

<u>Theorem 5:</u> subst[x;a;u]* subst[x;a;v] = subst[x;a;u*v]

In words, the operation of substitution is distributive over concatenation.

Proof: subst[x;a;u]*subst[x;a;v]

```
= [null[subst[x;a;u]] → subst[x;a;v]; T → cons[car[subst[x;a;u]];
                                          cdr[subst[x;a;u]]*subst[x;a;v]]]
= [null[atom[u] → [eq[u;a] → x; T → u]; T → cons[subst[x;a;car[u]];
   subst[x;a;cdr[u]]]] → subst[x;a;v]; T → cons[car[
   subst[x;a;u]];cdr[subst[x;a;u]]*subst[x;a;v]]]
 = [atom[u] → [eq[u;a] → [....]; T → [null[u] → subst[x;a;v]; T →
   cons[car[subst[x;a;u]];cdr[subst[x;a;u]]*subst[x;a;v]]]];
 T → [null[cons[subst[x;a;car[u]];subst[x;a;cdr[u]]]] → subst[x;a;v];
 T → cons[car[subst[x;a;u]];cdr[subst[x;a;u]]*subst[x;a;v]]]]
```

Now noting that 1) for true lists atom[u] implies null[u]
2) null[u] implies by convention ~eq[u;a]
3) ~null[u] implies ~null[cons[subst[x;a;car[u]]; subst[x;a;cdr[u]]]]

(perhaps 3) is too obvious to note!)

the above awesome expression reduces to:

```
= [null[u] → subst[x;a;v]; T → cons[car[subst[x;a;u]];
                                    cdr[subst[x;a;u]]*subst[x;a;v]]]
```

and applying Lemma 1, again remembering ~null[u] is equivalent to ~atom[u] for true lists:

```
= [null[u] → subst[x;a;v]; T → cons[subst[x;a;car[u]];
                               subst[x;a;cdr[u]]*subst[x;a;v]]]
```

This suggests the functional equation

```
f[x;a;u;v] = [null[u] → subst[x;a;v]; T → cons[subst[x;a;car[u]];f[x;a;cdr[u];
                                                         f[x;a;cdr[u];v]]]
```

Now examining subst[x;a;u*v]

```
 = subst[x;a;[null[u] → v; T → cons[car[u];cdr[u]*v]]]
 = [null[u] → subst[x;a;v]; T → subst[x;a;cons[car[u];cdr[u]*v]]]
```

but in the case ~null[U];car[u] = car[u*v]
cdr[u]*v = cdr[u*v]

so we have

```
 = [null[u] → subst[x;a;v]; T → subst[x;a;cons[car[u*v];cdr[u*v]]]]
```

but in the case ∼null[u], car[u] = car[u*v]
cdr[u]*v = cdr[u*v]

so we have

= [null[u] → subst[x;a;v]; T → subst[x;a;cons[car[u*v]; cdr[u*v]]]]

and since ∼null[u] implies ∼atom [u*v]; Lemma 2 yields

= [null[u] → subst[x;a;v]; T → cons[subst[x;a;car[u*v]];
subst[x;a;cdr[u*v]]]]

or = [null[u] → subst[x;a;v]; T → cons[subst[x;a;car[u]];
subst[x;a;cdr[u]*v]]]

which also satisfies the equation f[x;a;u;v]

Q. E. D.Theorem

<u>Theorem 6:</u> subst[x;a;rev[u]] = rev[subst[x;a;u]]

where rev[u] is the function, whose domain is again true lists, defined on page 26 of the previously mentioned paper by

rev[u] = [null[u] → NIL; T → rev[cdr[u]]* cons[car[u];NIL]]

Proof: subst[x;a;rev[u]]

= subst[x;a;[null[u] → NIL; T → rev[cdr[u]]*cons[car[u];NIL]]]

= [null[u] → subst[x;a;NIL]; T → subst[x;a;rev[car[u]]*cons[car[u];NIL]]]

= (by Lemma 4 and Theorem 5) [null[u] → NIL; T →
subst[x;a;rev[cdr[u]]]*subst[x;a;cons[car[u];NIL]]]

= (by Lemma 5) [null[u] → NIL; T → subst[x;a;rev[cdr[u]]]*
cons[subst[x;a;car[u]];NIL]]

Looking now at rev[subst[x;a;u]]

= [null[subst[x;a;u]] → NIL; T → rev[cdr[subst[x;a;u]]]*
cons[car[subst[x;a;u]];NIL]]

which by reasoning identical to that presented in great detail in the proof of Theorem 5 is equivalent to:

= [null[u] → NIL; T → rev[subst[x;a;cdr[u]]]*cons[
subst[x;a;car[u]];NIL]]

(using also Lemma 1)

Both expressions under consideration are then solutions of the functional equation

f[x;a;a] = [null[u] → NIL; T → f[x;a;cdr[u]]*cons[subst[x;a;car[u]];NIL]]

Q. E. D.Theorem 6

We now turn to the second major part of this paper. Having discovered the algebraic properties of subst[x;y;z], we now wish to

undertake an investigation on a slightly different level. In particular, we will consider the behavior of subst[x;y;z] on operations which are distributive over concatenation.

Firstly, the set of functions of true lists which are distributive over concatenation is non-empty.

Consider the function defined by:

sq[u] = [null[u] → NIL; T → cons[car[u]; cons[car[u];NIL]]* sq[cdr[u]]]

sq[u] might best be described by an example:

sq[(A,(B,C),C)]=(A,A,(B,C),(B,C),C,C)

Lemma 7: If ∼ null[u], then car[sq[u]] = car[u].

Proof: car[sq[u]]

= car[null[u] → NIL; T → cons[car[u];cons[car[u];NIL]]* sq[cdr[u]]]

= car[cons[car[u]; cons[car[u];NIL]]* sq[cdr[u]]]

= [null[cons[.....]] → ...]; T → car[cons[car[u];cons[car[u];NIL]]]]

= car[u]

Le Q. E. D.

Lemma 8: If ∼ ~~null~~[u], then cdr[sq[u]] = cons[car[u];NIL]*sq[cdr[u]].

Proof: cdr[sq[u]]

= cdr[null[u] → NIL; T → cons[car[u]; cons[car[u];NIL]]*sq[cdr[u]]]

= cdr[cons[car[u];cons[car[u];NIL]]* sq[cdr[u]]]

= [null[cons[...]] → ...]; T → cdr[cons[car[u];cons[car[u];NIL]]]* sq[cdr[u]]]

= cons[car[u];NIL]*sq[cdr[u]] Q. E. D.

Lemma 9: sq[cons[u;v]] = cons[u;cons[u;NIL]]*sq[v]

Proof: sq[cons[u;v]]

= [null[cons[u,v]] → ...; T → cons[car[cons[u;v]];cons[car[cons[u,v]];NIL]] *sq[cdr[cons[u;v]]]]

= cons[u;cons[u;NIL]]* sq[v] Q. E.D.

Before proceeding, we note that Theorem 11 of the previously mentioned paper of McCarthy proves the associativity of the operation of concatenation, and therefore the notation x * y * z introduces no ambiguity.

Theorem 7: sq[u*v] = sq[u]* sq[v]

Proof:

sq[u*v] = sq[null[u] → v; T → cons[car[u];cdr[u]*v]]

= [null[u] → sq[v]; T → sq[cons[car[u];cdr[u]*v]]]

= (by Lemma 9) [null[u] → sq[v]; T → cons[car[u];cons[car[u];NIL]]* sq[cdr[u]*v]]

and sq[u]*sq[v]

= [null[sq[u]] → sq[v]; T → cons[car[sq[u]];cdr[sq[u]]*sq[v]]]

= [null[null[u] → NIL; T → cons[...]] → sq[v]; T → cons[car[u];
cons[car[u];NIL]* sq[cdr[u]] * sq[v]]]

by Lemmas 7 and 8; and now since null[cons[...]] = F, this reduces to

= [null[u] → sq[v]; T → cons[car[u];cons[car[u];NIL]* sq[cdr[u]]*sq[v]]]

To be precise, we need a lemma:

Lemma 10: If ~null[u], then cons[x;u*v] = cons[x;u]*v.

Proof: cons[x;u]*v

= [null[cons[x;u]] → ..]; T → cons[car[cons[x;u]];cdr[cons[x;u]]*v]]

= cons[x;u*v]

Q. E. D. Lemma 10

Applying Lemma 10 in the proof of Theorem 7, we obtain:

sq[u]*sq[v]

= [null[u] → sq[v]; T → cons[car[u];cons[car[u];NIL]]*
sq[cdr[u]]*sq[v]]

and both forms satisfy the equation

f[u;v] = [null[u] → sq[v]; T → cons[car[u];cons[car[u];NIL]]*
f[cdr[u];v]]

Q. E. D. Theorem 7

Thus the function sq[u] is distributive over concatenation.

Instead of proving directly that subst[x;a;sq[u]] = sq[subst[x;a;u]], we will prove the following more general result:

Theorem 8: If f is a function whose domain and range are true lists, with the equation defining f containing no "constants," such that f[u*v] = f[u]*f[v], and such that
~[null[f[u]]∧~null[u]], then

subst[x;a;f[u]] = f[subst[x;a;u]].

(The assumption that the equation defining f contains no constants is discussed after the proof of the Theorem. It is imprecisely stated and undoubtedly open to criticism. Also, it is a hypothesis of a different nature than the others, for it is not used explicitly in the proof but rather to emphasize the exclusion of situations such as the one that will be discussed later in this paper.)

Proof: We first list five facts, some of which have been noted earlier, to which we will refer in the course of this proof:

a) ~null[a] is always assumed, by our convention about subst[x;a;u].

b) For true lists, atom[u] implies null[u].

c) With the hypotheses of this Theorem, we actually have the situation null[f[u]] if and only if null[u]. The case null[f[u]] only if null[u] is assumed directly; the converse follows from the distributivity of f over concatenation, for if null[u], then f[u] = f[NIL] = f[NIL*NIL] (since NIL*u = u follow immediately from the definition of concatenation) = f[NIL] * f[NIL] = f[u]*f[u], and this can only be true if we have null[f[u]]. Therefore we freely substitute null[u] for null[f[u]], and vice versa.

d) If ~null[u], then u = cons[car[u];NIL]*cdr[u]. This is so because cons[car[u];NIL]*cdr[u]

= [null[cons[...]] → ...; T → cons[car[cons[car[u];NIL]];
cdr[cons[car[u]; NIL]]* cdr[u]]]

= cons[car[u]; NIL * cdr[u]]

= cons[car[u]; cdr[u]]   since NIL*u = u,

= u

e) If ~null[u], then ~null[subst[x;a;u]] (u a true list).
This follows directly from the equation defining subst[x;a;u].

We now begin the actual proof: subst[x;a;f[u]]

= [atom[f[u]] → [eq[f[u];a] → x; T → f[u]]; T → cons[subst[x;a;car[f[u]]];
subst[x;a;cdr[f[u]]]]]

= [null[f[u]] → f[u]; T → cons[subst[x;a;car[f[u]]];subst[x;a;cdr[f[u]]]]]

by b) and a); we now apply d) to the argument of f in the expressions car[f[u]] and cdr[f[u]]. (We can do this by c), since we are in the case ~ null[f[u]].), and we obtain

= [null[f[u]] → f[u]; T → cons[subst[x;a;car[f[cons[car[u];NIL]*cdr[u]]]];
subst[x;a;cdr[f[cons[car[u]; NIL]* cdr[u]]]]]]

= [null[f[u]] → f[u]; T → cons[subst[x;a;car[f[cons[car[u];NIL]]*f[cdr[u]]]];
subst[x;a;cdr[f[cons[car[u];NIL]]*f[cdr[u]]]]]]

by the distributivity of f over concatenation,

= [null[f[u]] → f[u]; T → cons[subst[x;a;car[f[cons[car[u];NIL]]]];
subst[x;a;cdr[f[cons[car[u];NIL]]]* f[cdr[u]]]]]

by the identities for car[x*y] and cdr[x*y], which are applicable in this form because of the hypothesis on f, which insures that ~ null[cons[car[u];NIL]] (which is certainly true) implies ~null[f[cons[car[u];NIL]]]. Now, applying Theorem 5, we have

= [null[f[u]] → f[u]; T → cons[subst[x;a;car[f[cons[car[u];NIL]]]];
subst[x;a;cdr[f[cons[car[u];NIL]]]]*subst[x;a;f[cdr[u]]]]]

= [null[u] → f[u]; T → cons[car[subst[x;a;f[cons[car[u];NIL]]]];
cdr[subst[x;a;f[cons[car[u];NIL]]]]*subst[x;a;f[cdr[u]]]]]

by Lemma 1; we also substituted null[u] for null[f[u]], as allowed by c); but this is now seen to be

= [null[u] → f[u]; T → subst[x;a;f[cons[car[u];NIL]]]*subst[x;a;f[cdr[u]]]]

This last step follows because we had an expression of the form

cons[car[u];cdr[u]*v] with ~null[u] (by e), which was applicable by the hypothesis on f, as noted three steps previously)

= cons[car[u*v];cdr[u*v]] = u*v

= u*v

The expression for subst[x;a;f[u]] is now in the desired form, for we define a functional equation by

F[x;a;u] = [null[u] → f[u]; T → F[x;a;cons[car[u];NIL]]*F[x;a;cdr[u]]].

Now to work on f[subst[x;a;u]], which can be written

f[atom[u] → [eq[u;a] → x; T → u]; T →
    cons[subst[x;a;car[u]];subst[x;a;cdr[u]]]]

= [null[u] → [eq[u;a] → ..., T → f[u]]; T → f[cons[subst[x;a;car[u]]; subst[x;a;cdr[u]]]]]

by b) and a),

= [null[u] → f[u]; T → f[cons[subst[x;a;car[u]];subst[x;a;cdr[u]]]]].

Actually, we need only write this in the following form:

= [null[u] → f[u]; T → f[subst[x;a;u]]] justifying this seemingly backwards step by Lemma 1. Now, replacing u by subst[x;a;u] in d), which we can do by e), since we are in the case ~ null[u], we obtain

= [null[u] → f[u]; T → f[cons[car[subst[x;a;u]];NIL]*cdr[subst[x;a;u]]]]

= [null[u] → f[u]; T → f[cons[subst[x;a;car[u]];NIL]*subst[x;a;cdr[u]]]]

by Lemma 1,

= [null[u] → f[u];T → f[subst[x;a;cons[car[u];NIL]]*subst[x;a;cdr[u]]]]

by Lemma 5,

= [null[u] → f[u];T → f[subst[x;a;cons[car[u];NIL]]]*f[subst[x;a;cdr[u]]]]

by the distributivity of f over concatenation.

Thus f[subst[x;a;u]] also can be transformed into the required form satisfying the functional equation F[x;a;u], and the proof of Theorem 8 is complete.

Q. E. D.

The proof of Theorem 8, though lengthy, is a good example of the method and application of recursion induction. A few comments should be made about the particular content of this theorem and its proof:

1) The result is non-intuitive, at least more so than the other, more specific results of this paper.

2) The function sq[u] satisfies the hypotheses of Theorem 8, for

it is distributive over concatenation by Theorem 7, and null sq[u] implies null[u], as noted in the body of the proof of Theorem 7. Therefore we have the corollary: subst[x;a;sq[u]] = sq[subst[x;a;u]].

3. We must be careful about our definition of a function whose domain is true lists; sq[u] certainly suffices, but problems may arise. The next section of this paper is devoted to this topic.

Consider the simple minded function which inserts the list u before every element of a list v. For example, if (A,(A,B),C) were v, and u were (D,E), we would obtain

(D,E,A,D,E,(A,B),D,E,C).

Such a function could perhaps be considered as a function merely of the list v, for the list u is fixed. (The list u plays the role of the "constant" mentioned in the statement of Theorem 8.)"
As such, the function would satisfy the hypotheses of Theorem 8, for it will be seen (Theorem 9) that it is distributive over concatenation, and the condition $\sim$[null[f[u]] $\wedge\sim$null[u]] will be satisfied by the formulation of the definition of our function. Yet the conclusion of Theorem 8 is not valid in this case. (This will follow from Theorem 10.)

The reason is that the function is not merely a function on lists v but on lists u and v; i.e., its domain is the cartesian product of the space of true lists with itself.

The tale is told of the freshman who asked why we do not write f(3,x) = 3x instead of f(x) = 3x. The answer to his question is that 3x is merely an abbreviation for the operation x + x + x; or expressed in different words, the notation can be justified by the fact that polynomials in one indeterminate with coefficients in the real numbers indeed form a ring. In our example we have no such abbreviation, no such algebraic structure. Thus we cannot suppress the u, and we must define:

simf[u,v] = [null[v] → NIL; T → u*cons[car[v];NIL]*simf[u;cdr[v]]]

as our function. We will assume u is also a true list, and $\sim$null[u].

We will now prove the Theorems 9 and 10 referred to above. First we need three lemmas.

Lemma 11: If $\sim$null[v], then car[simf[u;v]] = car[u]

Proof: car[simf[u;v]] =
= car[null[v] → ..., T → u * cons[car[v], NIL] * simf[u;cdr[v]]]
= car[u * ....] = car[u] (since $\sim$null[u] by convention)

Q. E. D.

<u>Lemma 12:</u> If $\sim$ null[v], then cdr[simf[u;v]] =

Lemma 12: If ∼null[v], then cdr[simf[u;v]] =
cdr[u]*cons[car[v];NIL]*simf[u;cdr[v]]

Proof: cdr[simf[u;v]] =
= cdr[null[v] → ..., T → u*cons[car[v],NIL]*simf[u;cdr[v]]]
= cdr[u*cons[car[v];NIL]*simf[u;cdr[v]]]
= cdr[u]*cons[car[v];NIL]*simf[u;cdr[v]]

Q. E. D.

Lemma 13: simf[u;cons[v;w]] = u* cons[v;NIL]*simf[u;w]

Proof: simf[u;cons[v;w]]
= [null[cons[...]] → ...; T → u * cons[car[cons[v;w]];NIL]
* simf[u;cdr[cons[v;w]]]]
= u * cons[v;NIL]*simf[u;w] Q. E. D.

Theorem 9: simf[u;v*w] = simf[u;v]*simf[u;w]

Proof: simf[u;v*w]
= simf[u;[null[v] → w; T → cons[car[v];cdr[v]*w]]]
= [null[v] → simf[u;w]; T → simf[u;cons[car[v];cdr[v]*w]]]
= [null[v] → simf[u;w]; T → u * cons[car[v];NIL]*simf[u;cdr[v]*w]]
by Lemma 13;

and simf[u;v]*simf[u;w]
= [null[simf[u;v]] → simf[u;w]; T → cons[car[simf[u;v]];
cdr[simf[u;v]]*simf[u;w]]]
= [null[null[v] → NIL; T → u*...] → simf[u;w]; T → cons[car[u];
cdr[u]*cons[car[v];NIL]*simf[u;cdr[v]]*simf[u;w]]]
by Lemmas 11 and 12; and since ∼null[u] by convention,
= [null[v] → simf[u;w]; null[u*...] → ...; T → u*cons[car[v];NIL]
*simf[u;cdr[v]]*simf[u;w]]
since we had an expression of the form cons[car[u*v];cdr[u*v]],
= [null[v] → simf[u;w]; T → u*cons[car[v];NIL]*simf[u;cdr[v]]*simf[u;w]]

Thus both sides satisfy the functional equation
f[u;v;w] = [null[v] → simf[u;w];T → u*cons[car[v];NIL]*f[u;cdr[v];w]]

This establishes the distributivity of this insertion operation over concatenation. Q. E. D.

Theorem 10: subst[x;a;simf[u;v]] = simf[subst[x;a;u];subst[x;a;v]]

Proof: subst[x;a;simf[u;v]]
= subst[x;a;[null[v] → NIL; T → u*cons[car[v];NIL]*simf[u;cdr[v]]]]
= [null[v] → subst[x;a;NIL]; T → subst[x;a;u*cons[car[v];NIL]*simf[u;cdr[v]]]]
= [null[v] → NIL; T → subst[x;a;u]*subst[x;a;cons[car[v];NIL]]*
subst[x;a;simf[u;cdr[v]]]]
by Lemma 4 and repeated application of Theorem 5.

This suggests the functional equation:

f[x;a;u;v] = [null[v] → NIL; T →subst[x;a;u]*subst[x;a;cons[car[v];NIL]]
* f[x;a;u;cdr[v]]]

We have simf[subst[x;a;u];subst[x;a;v]]
= [null[subst[x;a;v]] → NIL; T → subst[x;a;u]*cons[car[subst[x;a;v]]; NIL]
NIL]*simf[subst[x;a;u];cdr[subst[x;a;v]]]]
= [null[v] → NIL; T → subst[x;a;u] * cons[subst[x;a;car[v]];NIL]
* simf[subst[x;a;u];subst[x;a;cdr[v]]]]

by reasoning analoguus to that presented in detail in Theorem 5, and by Lemma 1;

= [null[v] → NIL; T → subst[x;a;u]*subst[x;a;cons[car[v];NIL]]
* simf[subst[x;a;u];subst[x;a;cdr[v]]]]

by Lemma 5.

This also satisfies the functional equation.

Q. E. D.
Theorem 10

This completes the work of this paper. One comment alone remains, and that concerns convergence of the functional equations defined. In this paper they are all of two forms:

1) (u an S-expression) f[...;u;..] =[atom[u] →...,T→g[...;f[...;car[u];1]
g[...;f[...;car[u];...];f[...;cdr[u];...];...]...];
f[...;cdr[u];...];...]]

2) (u a true list) f[...;u;..] = [null[u]→...;T → g[...;f[...;cdr[u];..]
g[...;f[...;cdr[u];...];...]]
...]]

where the other arguments of g are well defined specific functions which, if they are recursive functions, are known to converge under the conditions ~atom[u] or ~ null[u] respectively. g itself is a specific function which converges under these conditions as long as each of its arguments is well defined. Thus it is the construction (or definition) of S-expressions and true lists which in the long run insures the convergence of the functional equations. Any need for a more rigorous formulation of convergence for the purposes of this paper is doubtful, but one must be aware of the problem of convergence whenever working with recursion induction.

January 19, 1962
Revised: March 6, 1962

CS-TR Scanning Project
Document Control Form

Date : 11 / 30 / 95

Report # AIM - 37

Each of the following should be identified by a checkmark:
Originating Department:

- [x] Artificial Intellegence Laboratory (AI)
- [ ] Laboratory for Computer Science (LCS)

Document Type:

- [ ] Technical Report (TR)
- [x] Technical Memo (TM)
- [ ] Other: ____________

Document Information

Number of pages: 16(20-images)
Not to include DOD forms, printer intstructions, etc... original pages only.

Originals are:
- [ ] Single-sided or
- [x] Double-sided

Intended to be printed as :
- [ ] Single-sided or
- [x] Double-sided

Print type:
- [ ] Typewriter
- [ ] Offset Press
- [ ] Laser Print
- [ ] InkJet Printer
- [ ] Unknown
- [x] Other: POOR COPY OF MIMEOGRAPH

Check each if included with document:
- [ ] DOD Form
- [ ] Funding Agent Form
- [ ] Cover Page
- [ ] Spine
- [ ] Printers Notes
- [ ] Photo negatives
- [ ] Other: ____________

Page Data:

Blank Pages(by page number): ____________

Photographs/Tonal Material (by page number): ____________

Other (note description/page number):

Description :	Page Number:
IMAGE MAP! (1-16) UNH TITLE & BLANK	1-14
(17-20) SCANCONTROL, TRGT'S (3)	

Scanning Agent Signoff:
Date Received: 11 / 30 / 95 Date Scanned: 12 / 11 / 95 Date Returned: 12 / 14 / 95

Scanning Agent Signature: Michael W. Cook

Rev 9/94 DS/LCS Document Control Form cstrform.vsd

Scanning Agent Identification Target

Scanning of this document was supported in part by the **Corporation for National Research Initiatives,** using funds from the **Advanced Research Projects Agency** of the **United states Government** under Grant: **MDA972-92-J1029.**

The scanning agent for this project was the **Document Services** department of the **M.I.T Libraries.** Technical support for this project was also provided by the **M.I.T. Laboratory for Computer Sciences.**